AMENDMENT AND PRESENTATION OF CLAIMS

Please replace all prior claims in the present application with the following claims, in which claims 1, 11 and 20 are currently amended, and claim 21 is newly presented.

1. (Currently Amended) A distributed system for process automation~~; said distributed system~~ comprising:

~~a computer readable medium, an event router component, a means to start component, a conductor component and a plurality of distributed services agents,~~ -

~~said~~ a computer-readable medium ~~storing~~ configured to store a representation of the process as a plurality of nodes and a plurality of directed links, each said node having a plurality of attributes, each of said directed links connecting two of said nodes and representing the sequencing and possibly data dependency between the two of said nodes, said plurality of nodes including a plurality of start nodes;

~~said~~ an event router component ~~receiving~~ configured to receive status transition events for said nodes and dispatching said events to said conductor and distributed services agents components;

~~said~~ means to start component for producing 'ready2go' events for said plurality of start nodes~~,~~ ;

~~said~~ a conductor component ~~handling~~ configured to handle 'ready2go' and 'completed' events by generating 'inprocess' events in response to 'ready2go' events and deciding when to produce additional 'ready2go' events for successor nodes in response to 'completed' events~~,~~ ; and

~~said~~ a distributed services agent ~~handling~~ configured to handle 'inprocess' events by performing application work and producing 'completed' or 'error' events.

2. (Original) The distributed system as recited in claim 1, wherein said representation is configured to define non-overlapping sections of the process, and having at most as many active instances of the process as said non overlapping sections of the process,

wherein each of said instances of the process having at most one non-overlapping section that has nodes in statuses other than 'notreached' or 'completed', and nodes in other sections in the same 'notreached' or 'completed' status.

3. (Original) The distributed system as recited in claim 1, wherein said representation is configured to specify cached data in a node, such that a successor node to said node starts execution further when the status of said node is 'notreached'.

4. (Original) The distributed system as recited in claim 1, further comprising said representation is configured to define sub-flows, and a means to invoke a plurality of instances of said sub-flows.

5. (Original) The distributed system as recited in claim 1, further comprising said representation is configured to define an error handling flow, and invoke one instance of said error handling flow when one node in said plurality of nodes receives an 'error' event.

6. (Original) The distributed system as recited in claim 1, further comprising a means to evaluate at run time said node attributes and using said evaluated attributes when performing said application work.

7. (Original) The distributed system as recited in claim 5, wherein said means to evaluate at run time said node attributes comprises computing parameters and having a means to handle parameter clashes.

8. (Original) The distributed system as recited in claim 1, further said representation is configured to specify conditional dependencies, and said conductor component recognize a `pseudo-completed` event for nodes transitioned to via non satisfied said conditional dependencies.

9. (Original) The distributed system as recited in claim 1, further comprising a repository component that stores the event history, and a means to play back the process execution based on said event history.

10. (Original) The distributed system as recited in claim 1, further comprising a plurality of domains, wherein each of said domains has a plurality of processes out of which at most one is active at any time.

11. (Currently Amended) A method for dataflow automation in a system comprising a plurality of data management systems, comprising:

storing a representation of the dataflow having a plurality of nodes and links, wherein each of the nodes corresponds to a process managing a portion of the dataflow in one of the data management systems, and each of the links connect two of the nodes and correspond to a dependency between two of the processes that correspond to the two of the nodes, wherein the data management systems are heterogeneous systems;

sensing an event that occurs in one of the processes managing the dataflow in one of the data management systems, wherein the sensed event indicating whether the one of the processes has completed or has produced an error; and

in response to the sensed event, scheduling a task to manage data portion of the dataflow in another process in another of the data management system based whether events have been received indicating that predecessors for the other process indicated in the representation has completed or has produced an error.

12. (Original) A method for dataflow automation according to claim 11, further comprising:

tagging some of the nodes in the representation as boundary nodes that define a plurality of sections of the dataflow, wherein the scheduled task is operating in one of the sections; and

scheduling another task in another section of the dataflow, wherein the scheduled tasks and the other scheduled task are active at the same time.

13. (Original) A method for dataflow automation according to claim 11, further comprising:

defining, in the representation, a subroutine of sub-flows that specifies a set of processes to be executed based on an invoking node; and executing the subroutine of sub-flows at an invoking node.

14. (Original) A method for dataflow automation according to claim 13, wherein the subroutine of sub-flows includes an error handler subroutine.

15. (Original) A method for dataflow automation according to claim 11, further comprising:

storing an attribute for each node in the representation as a string that references one or more parameters produced by predecessor nodes of the node; and substituting the parameters in the attribute when invoking a process corresponding to the node.

16. (Original) A method for dataflow automation according to claim 15, further comprising:

determining whether at least two of the predecessor nodes have produced a parameter with a same name; and selecting a value of the most recent parameter when at least two of the predecessor nodes have produced a parameter with a same name.

17. (Original) A method for dataflow automation according to claim 11, further comprising:

selecting a successor process from among a plurality of successor processes at node to schedule based on a condition;

scheduling a task for the selected successor process; and

marking other of the successor process with one of a status indicating a completion.

18. (Original) A method for dataflow automation according to claim 11, further comprising: recording events and invocation of processes in a repository; and playing back the recorded events and invocation of processes.

19. (Original) A method for dataflow automation according to claim 11, further comprising: defining a plurality of domains; and executing a separate dataflow in each of the domains.

20. (Currently Amended) A computer-readable storage medium bearing instructions for dataflow automation, said instructions being arranged, upon execution by one or more processors, to perform the method according to claim 11.

21. (New) A method of data processing, the method comprising:

receiving information, corresponding to a process, from a plurality of heterogeneous data platforms; and

providing a sense-and-response framework to process the received information, wherein the framework receives events in the process and initiates a plurality of heterogeneous tasks in response to the events,

wherein the tasks are executed until the process reaches an endpoint.